4992–4999 Nucleic Acids Research, 1995, Vol. 23, No. 24

A new DNA sequence assembly program

James K. Bonfield, Kathryn F. Smith and Rodger Staden*

MRC Laboratory of Molecular Biology, Hills Road, Cambridge CB2 2QH, UK

Received September 29, 1995; Revised and Accepted November 15, 1995

ABSTRACT

We describe the Genome Assembly Program (GAP), a new program for DNA sequence assembly. The program is suitable for large and small projects, a variety of strategies and can handle data from a range of sequencing instruments. It retains the useful components of our previous work, but includes many novel ideas and methods. Many of these methods have been made possible by the program's completely new, and highly interactive, graphical user interface. The program provides many visual clues to the current state of a sequencing project and allows users to interact in intuitive and graphical ways with their data. The program has tools to display and manipulate the various types of data that help to solve and check difficult assemblies, particularly those in repetitive genomes. We have introduced the following new displays: the Contig Selector, the Contig Comparator, the Template Display, the Restriction Enzyme Map and the Stop Codon Map. We have also made it possible to have any number of Contig Editors and Contig Joining Editors running simultaneously even on the same contig. The program also includes a new 'Directed Assembly' algorithm and routines for automatically detecting unfinished segments of sequence, to which it suggests experimental solutions.

INTRODUCTION

Good software is essential to ensure the accuracy of the data produced by DNA sequencing projects and, as it has a major influence on the efficiency and user time required to complete the sequence, it also plays an important role in determining the overall cost of projects. Many programs and algorithms for handling sequencing projects have been published (1–8). Our assembly software has been evolving on several fronts for a number of years and the predecessors of the program described here are being used in some of the largest genome projects. While continuing to support our previous programs we have been working to put into effect the experience gained through this period of collaboration and the resulting Genome Assembly Program (GAP) is described here. The novel displays and methods of interaction contained in GAP should further increase the efficiency of genome projects. For example the new graphical overviews, and the visual clues they contain, plus improved editing capabilities, should make it much easier for users to interpret and check difficult assemblies that contain many repeats. The program also includes several functions for finding 'problems' and suggesting possible experimental solutions.

The program contains too many functions and modes of operation to be fully covered in the available space so we have been selective in the topics described and also have concentrated on outlining what the program can do for users, rather than the programming methods by which it is achieved. For the same reason, all but two of the figures shown have been reduced to less than half of the size that they would occupy on the user's screen, and some of their utility may have been lost in this process.

METHODS

The algorithms are written in ANSI C and FORTRAN 77, and the user interface is written in Tcl and Tk, which in their current versions produce a Motif style 'look and feel'. The program is being used with X windows on Sun, DEC and SGI UNIX workstations. One of the main design goals was that the algorithms should produce results that were made available to viewing methods that are under the user's control. The user should be able to choose from a variety of ways of looking at the data and should be able to control which items were visible and when they were deleted. The program uses several alignment algorithms (9,10).

RESULTS

First we cover the data sources, file formats and data types used by the program, then we deal briefly with some of the algorithmic aspects. The rest of the paper deals with the displays produced by the algorithms and the ways the user can interact with them.

Data sources and file formats

The programs can handle data produced by sequencing instruments such as ABI 373A and 377, the Pharmacia A.L.F., and the LiCor. They can also use data entered using digitizers (11) or that has been typed by hand. Usually the trace data files which are in proprietary formats are converted to SCF format files (12). As most of the instruments do not yet provide numerical estimates of the accuracy of each called base we calculate our own values from the ratios of the areas of superimposed peaks (13). All these preassembly steps plus quality clipping, sequencing vector and cosmid vector removal are controlled by the script PREGAP (14). During this processing the readings are stored in 'Experiment Files' (14).

* To whom correspondence should be addressed

Experiment File format is similar to that of EMBL sequence library entries in that each record starts with a two-letter identifier, but we have invented new records specific to sequencing experiments. The two-letter record identifiers make it very easy to parse the files and to write scripts to process the data. PREGAP can augment the files to include information about the vectors, primers and templates used in their production and, if necessary, can extract this information from external databases. Some of the information is needed by the preassembly programs, and some by GAP. The assembly program database stores the readings plus sufficient data about them and how they were produced to enable it to perform all the operations described in this article. The only external data used are the trace files, and their names are stored in the database so that traces can be displayed from within the editors. Note that the segments of sequence from the ends of machine-read data that are judged to be of too low quality to be included in the consensus, or that are found to be vector, are referred to as 'hidden' data. Hiding the poor quality data aids the assembly algorithm and reduces the amount of editing required. However, see 'Experiment suggestion functions'.

Annotating and labelling readings and contigs

A very useful and versatile feature of the program is its facility for labelling segments of readings and consensus sequences using 'tags'. The program recognises the set of standard tag types shown in Table 1, and users can also invent their own. Each tag type has a unique four character identifier, a name, a direction, a colour and a text string for recording notes. Tags can be created, edited and removed by users and by internal routines. Tags can also be input along with readings. For example all readings can be screened for Alu segments prior to assembly by the Alu search program REPE (14) which adds an appropriate record to the reading's experiment file. This information is copied into the database and becomes a tag attached to the reading.

Table 1. The standard tag types and their functions

Code	Function
COMM	Comment
COMP	Compression
RCMP	Resolved compression
STOP	Stop
OLIG	Oligo (primer)
REPT	Repeat
ALUS	Alu sequence
SVEC	Sequencing vector
CVEC	Cloning (say cosmid) vector
MASK	Mask me
FNSH	Finished segment
ENZ0	Restriction enzyme 0

Active tags and masking

Tags are used for a variety of purposes and, for each function in the program, the user can choose which tag types are currently 'active'. Where they are being used to provide visual clues this will determine which types of tag appear in the displays but, for other functions, they can be used to control which parts of the sequence are omitted from processing. This mode of operation is known as 'masking'. For example the program contains a routine to search for repeats and, if any are found, the user needs to know if such sequence duplications are caused by incorrect assembly or are genuine repeats. Once the user has checked a duplication reported by the program, and found it to be a genuine repeat, it can be labelled with a REPT tag. If the repeat search is run in masking mode and with REPT tags active, any segment covered by a REPT tag will not be reported as a match. So once the 'problem' has been attended to it can be labelled so it is not reported on subsequent searches. In addition the tag is available to provide annotation for the completed sequence when it is sent to the data libraries.

A more complicated application of masking is available for two of the other search procedures in the program: 'Normal Shotgun Assembly' and 'Find Internal Joins'. The former is the general assembly algorithm used in the program and the latter is used to find potential joins between the contigs in the database. Here we describe how masking can be used during assembly, and similar comments apply to Find Internal Joins. In the assembly function the user can choose to employ masking and then select the types of tags to be used as masks. Readings are compared in two stages: first the program looks for exact matches of some minimum length and then for each possible overlap it performs an alignment. If the masking mode is selected the masked regions are not used during the search for exact matches, but they are used during alignment. The effect of this is that new readings which would lie entirely inside masked regions will not produce exact matches and so will not be entered. However readings that have sufficient data outside of masked areas can produce matches and will be correctly aligned even if they overlap the masked data. A common use for masking during assembly or Find Internal Joins is to avoid finding matches that are entirely contained in Alu segments.

The consensus calculation

There are four main types of consensus sequence, including a 'quality' sequence that can be output by the program. The arithmetic performed uses the numerical estimates of base accuracy that are stored in the database (13), and the output formats include FASTA (15) and Experiment Files (14). Again the facility to use active tags is available, so tagged regions of the standard consensus can be shown in a special character set. For the masked character set we use d,e,f,i which are respectively equivalent to a,c,g,t. If the output is in Experiment File format active tags can be included in the file. An 'extended' consensus includes 'hidden' data from the ends of the contigs. It is employed by the internal search 'Find Internal Joins' and can be used for database searches. An 'unfinished' consensus is one which consists of a,c,g,t for single-stranded regions and d,e,f,i for finished sequence. These files could be used for screening purposes, i.e., only single-stranded regions would produce hits. A 'quality' consensus consists of the set of characters shown in Table 2. Here the consensus calculation is performed separately for each of the two strands of the data, and then the two are compared to produce the possibilities and codes shown in the table. For example 'c' means bad data on one strand is aligned with good data on the other.

Table 2. The quality code symbols and colours

Strand 1	Strand 2	Code	Colour
Good	Good (equal)	a	Blue
Good	Bad	b	Red
Bad	Good	c	Green
Good	None	d	Red
None	Good	e	Green
Bad	Bad	f	Yellow
Bad	None	g	Yellow
None	Bad	h	Yellow
Good	Good (disagree)	i	Black
None	None	j	Yellow

The codes assigned depend on the coverage (None, Bad or Good data) for each strand of the sequence.

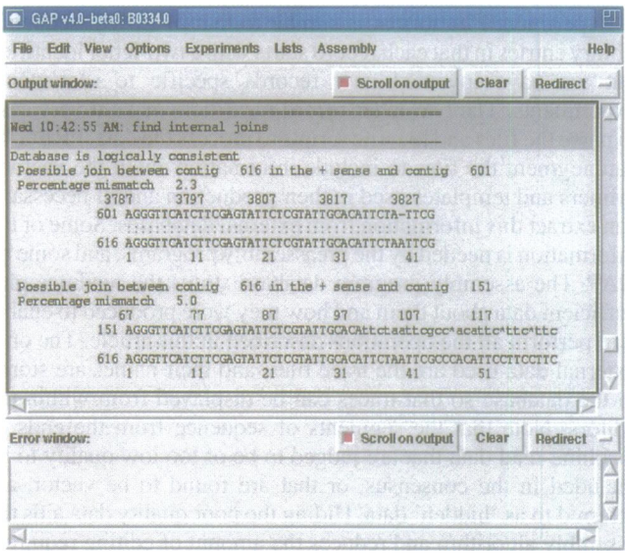

Figure 1. The main window of GAP.

Normal shotgun assembly

This is the mode that most users will employ for assembly. It takes one reading at a time and compares it with all the data already assembled in the database. If a reading matches, it is aligned. If the alignment is good enough the reading is entered into the database. If the reading aligns well with two contigs it is entered into one of them, then the two contigs are compared. If they align well they are joined. If the reading aligns well in more than two places the two best alignments are used. If the reading does not match it starts a new contig. If a reading matches but does not align well it can either be entered as a new contig or rejected. As mentioned above masking can be applied.

Assembly into single-stranded regions

This mode works like normal assembly with masking, except that the masking is not defined using tags, but occurs automatically for regions that already have sufficient data on both strands of the sequence. This means that new readings will only be assembled into regions that are single-stranded or which border, and overlap, such segments.

Directed Assembly

For Directed Assembly the Experiment File for each reading must contain a special 'Assembly Position' or AP line that defines the position at which to assemble the reading. The position is not defined absolutely, but relative to any other reading (the 'anchor reading') that has already been assembled. The definition includes the name of the anchor reading, the sense of the new reading, its offset relative to the anchor reading and the tolerance, i.e.:

AP anchor_reading sense offset tolerance

e.g.:

AP fred.021 + 1002 40

The sense is defined using + or – symbols, the offset can be of any size and can be positive or negative. For normal use 'tolerance' is a non-negative value, and the first base of the new reading must be aligned within +/– 'tolerance' bases of 'offset'. If 'tolerance' is zero, after alignment the position must be exactly 'offset' relative to the anchor reading. If 'tolerance' is negative then alignment is not performed and the reading is simply entered at position 'offset' relative to the anchor reading. If the anchor reading is named *new* the reading starts a new contig and the other values on the AP line are not required.

The algorithm is as follows: get the next reading name, read the AP line, find the anchor reading in the database, get the consensus for the region defined by anchor_reading + offset +/– tolerance. Perform an alignment with the new reading, check the position and the percentage mismatch. If OK enter the reading.

Breaking and disassembling contigs

Sometimes it is necessary drastically to alter contigs, and GAP contains routines for breaking contigs, disassembling contigs, moving readings to other contigs and removing readings from the database.

Experiment suggestion functions

GAP contains three functions that can analyse contigs to find regions that require further data, then suggest appropriate experiments to be performed, and the templates to use. The types of experiments suggested depend on the currently available technology. GAP can find single-stranded regions which need filling (this will also generally include the ends of contigs, and hence means that the experiment will extend the contig) and either suggest readings to resequence using the 'long gel' instruments that are now available, or will select a primer and template for a new reading. By searching the database for COMP and STOP tags (Table 1) the program can also find the names of readings to be resequenced using special methods (16,17) that might resolve such problems. The lists produced by the suggestion functions can be used to send requests for the relevent experiments to be performed. A related function 'Double Strand' fills single-stranded regions of contigs with the hidden data from neighbouring readings. That is, hidden data which align well are changed to visible, and will in future be included in the consensus calculation.

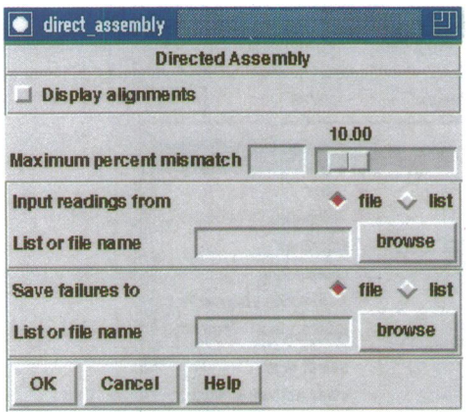

Figure 2. A Directed Assembly dialogue panel.

Overview of the user interface

The top of the Main Window contains the menubar; underneath and attached to this is the Output Window and underneath that is the Error Window. The GAP commands are accessed through the main menu. Each command brings up a single dialogue window containing all the relevant questions. An example of the Main Window is shown in Figure 1, and an example dialogue window, that for Directed Assembly, is shown in Figure 2. Several of the functions will produce results that are displayed in the Output Window and plotted in the Contig Comparator; others, such as the Template Display, create new windows.

The graphical displays have several common features. Commands are selected from buttons and menus at the top of the window. Menus can be 'torn off' and positioned anywhere on the screen. 'Zoom in' and 'Zoom out' buttons change the magnification of the plot in response to the sizes of areas of the display dragged out using a mouse button. 'Back' incrementally returns to previous levels of magnification. Cross hairs and cursors can be toggled on and off, and their coordinates in base positions appear in boxes in the top right-hand corner of the displays. At the bottom of the displays is an information line. Every item plotted in the graphical displays has text attached, and as the cursor passes over an item it is highlighted and its text appears in the information line.

The Contig Selector and Contig Comparator

The Contig Selector displays all the contigs in the database as colinear horizontal lines separated by short vertical lines. The length of the horizontal lines is proportional to the length of the contigs, and their left-to-right order represents the current ordering of the contigs. The order is stored in the GAP database and the user can change it by dragging the contig lines within the display. In Figure 3, the Contig Selector is the part of the display above the large, square box. As can be seen, the active tags for the contigs can be displayed: those above the lines are on readings and those below are on the consensus. If the user clicks on a tag, information about it will appear in the Output Window. The tags provide landmarks within the sequence and, as this display is always visible when a database is open, it gives a continuous overview of the state of the project. The Contig Selector can be

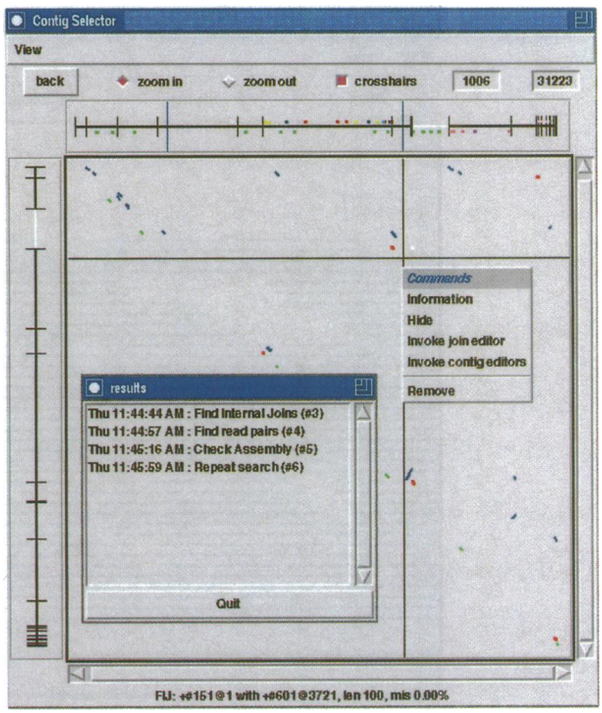

Figure 3. The Contig Comparator.

used to obtain information about each of the contigs and to select contigs for processing.

Several analytical functions within the program automatically transform the Contig Selector to produce the Contig Comparator (Figure 3). For this plot a copy of the Contig Selector is added at right angles to the original window so as to create a two dimensional rectangular surface on which to display the results of comparing contigs, or of possible problems in contigs. It can currently plot the results of searches for templates that have readings in more than one contig (Find Read Pairs), the results of searches for possible overlaps between contigs (Find Internal Joins) and searches for repeats (Find Repeats), plus the results from searches for poor alignments (Check Assembly). Each analytical function produces matches that are plotted as diagonal lines in different colours.

Figure 3 contains the results of Find Read Pairs in blue, Find Internal Joins in red and Check Assembly in green. Notice that at the top left of the display are several results plotted in blue close to, and in parallel with, the main diagonal. This indicates that the contigs in that region are in the correct relative orientations and order to one another. Points plotted away from the main diagonal are produced by pairs of contigs that are not in the correct relative order. Those plotted perpendicular to the main diagonal correspond to pairs of contigs in opposite orientations. The manual contig dragging procedure outlined above can still be performed, and the plotted results associated with any contig will move along with it to its new location in the display. Dragging the contigs to move the plotted match results close to the main diagonal will simultaneously put the contigs into the correct relative positions.

In Figure 3 the Contig Comparator is shown with the Results Manager window superimposed in the bottom left corner. This shows the results currently available and the time they were created. Clicking on a record in this window causes a pop-up

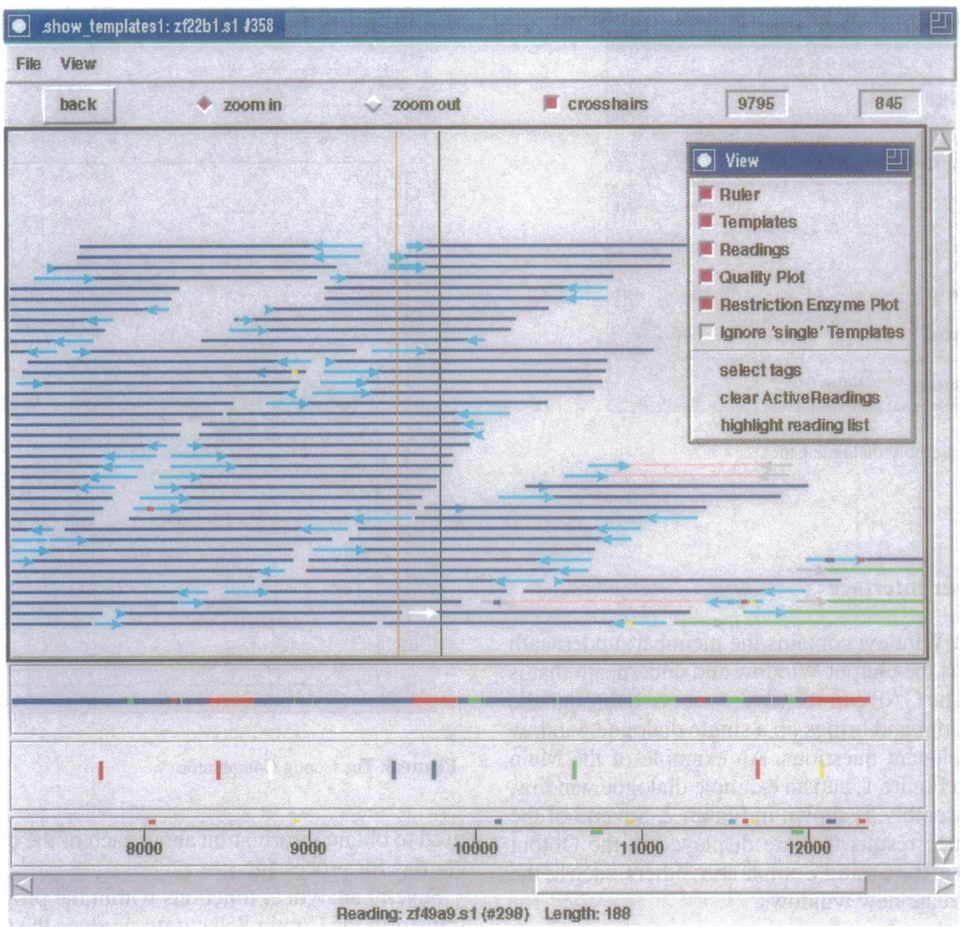

Figure 4. The Template Display.

menu to appear from which operations can be selected, such as changing the colour used for plotting, to perform on the corresponding set of results. Just above the Results Manager window is a cluster of three points. The leftmost one is from Find Internal Joins and the one to its right from Find Read Pairs. Their close proximity indicates that they are from the same pair of contigs and hence provide independent but corroborating evidence for an overlap between the two contigs. The dot plotted just below these two is a result from Check Assembly. These results always appear on the main diagonal and they show the positions of readings which do not fit well with the readings they overlap. This display allows their positions relative to the results from repeat searches to be seen and hence shows if they should be moved to another location.

The Contig Comparator can also be used to invoke the Join Editor, the Contig Editor and the Template Displays. To do this the user simply clicks on the plotted result of interest and a pop-up menu appears from which options can be selected (Figure 3). If the Join Editor is selected the editor will start up with the two contigs in the correct relative orientation (i.e., if necessary it will complement contigs), and aligned at the matching position. If the Template Display is selected from a plotted match produced by Find Read Pairs, then two template displays will appear with the readings from the template that spans the two contigs highlighted.

The Template Display

Figure 4 shows a typical view of the Template Display with a 'View' menu superimposed. As indicated by the active items in the View menu the user has elected to show all the templates, readings, quality plot and restriction enzyme sites. (Note that optionally, the templates with single readings can be hidden.) Some tag types are also active as they can be seen above (those on readings) and below (those on the consensus) the ruler, and also on the lines representing the readings. In the main part of the display templates are shown in dark blue (those with single readings), pink (those with readings taken from both ends) and green (those with readings on each end, one of which is in another contig, hence indicating a possible join). Grey is used for readings in the reverse sense and light blue for those in the forward sense.

The black vertical line is the Template Display cursor. The orange vertical line shows the position of the cursor in the Contig Editor, and the editor can be scrolled by dragging this line with the mouse. Below this section of the display is a horizontal line showing the 'Quality Plot' which, using colour codes of Table 2, indicates the type of data coverage for each character in the consensus. In the next horizontal panel short vertical lines show the positions of restriction sites found by the program.

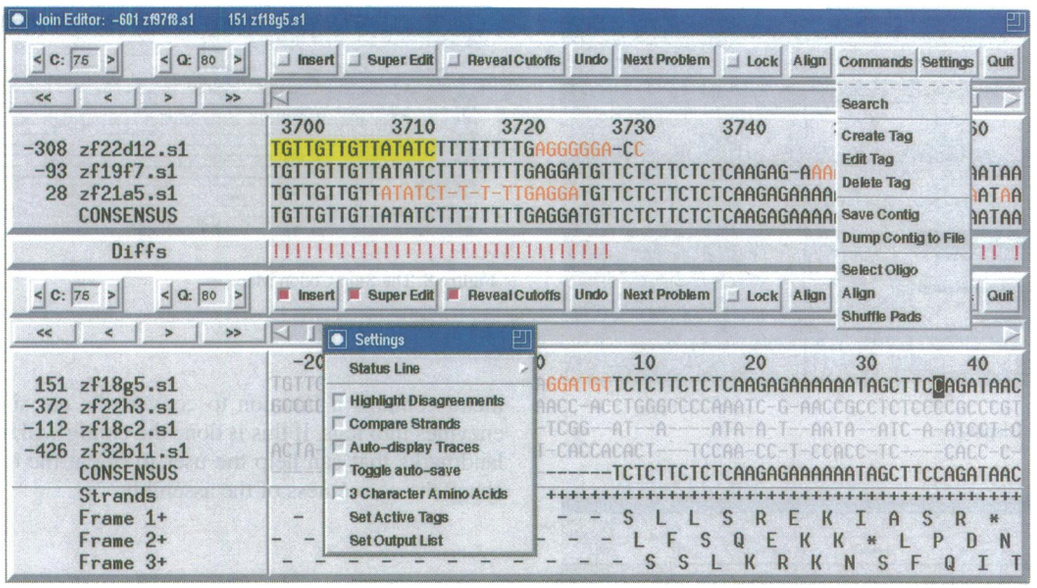

Figure 5. The Join Editor.

The Contig Joining Editor and the Contig Editor

Figure 5 is a screen dump of the Contig Joining Editor chosen to illustrate features of the program rather than to show a typical selection of settings. The left contig is shown in the top panel and the right contig below. Each display includes the reading numbers, names and sequences that cover a segment of the contig. Readings in the complementary sense have negative numbers. Underneath each contig is the consensus derived from the readings aligned above. Between the two contigs is a row marked 'Diffs' in which differences in the two consensus sequences are marked by '!' symbols. The user can edit the two contigs independently and then join them to form a single contig.

In this example the sequence data are shown in four different colours. Good quality data above the accuracy cutoff of 80 (see the box near the top left of the display) are shown in black and data below this cutoff in red. In the bottom contig the user has elected to reveal the 'cutoff' data from the ends of the readings. These are of poor quality or vector sequence and hence normally hidden, but when revealed are shown in grey. The yellow segment indicates the position of a tag. Beneath the lower contigs' consensus is a line of symbols showing information about the readings aligned above. This status line is being used here to indicate if the sequence is double stranded (=), reverse sense only (–) or, as here, positive sense only (+). Below this is a three phase amino acid translation of the consensus. Other displays are also available.

Along the top of each of the two contig displays are a series of buttons and menus which can be used independently for each of the contigs. At the left are two boxes that show the current consensus cutoff 'C' and the current accuracy estimate cutoff 'Q'. These values can be altered using the buttons to either side or by typing into the boxes. 'Super Edit' switches off several restrictions which are normally active to protect the data from inexpert users. 'Undo' incrementally reverses each previous editing operation. 'Next Problem' moves the cursor to the next place in the data that requires attention. This function works in several modes and can also automatically display the trace for the problem reading plus those for the best forward and reverse readings covering the same point. (The program selects the best readings by analysing the accuracy estimates for the readings, 13). The 'Lock' button forces the two contigs to scroll in register even though they are edited independently. 'Align' is used to align the two contig consensus sequences.

The 'Search' option in the 'Commands' menu creates a new window which offers a variety of searching functions. Search types and items include: position, problem, annotation, sequence, quality, reading name, edit, verify and tag type. 'Annotation' searches the comment fields of tags for a given text string. 'Quality' finds the next base that is not covered by data on both strands of the sequence. 'Edit' finds the position of the next base that has been edited. 'Verify' finds bases in the consensus for which there is no evidence in any of the original readings that cover it.

In the 'Commands' menu 'Dump Contig to File' creates a file containing a printable form of all the aligned readings in the contig. 'Select Oligo' selects a primer for a segment of the contig using the algorithm OSP (18) and creates a tag for it. 'Align' aligns a segment of a reading with the consensus. 'Shuffle Pads' aligns padding characters.

In the 'Settings' menu the 'Status Line' is a cascading menu for selecting combinations of data to display below the consensus sequence. 'Highlight Disagreements' toggles between the current display of readings and an alternative in which only differences between readings and the consensus are shown. 'Compare Strands' toggles the consensus sequence to one for which single-stranded regions are written as '–' characters, the rest being written as A,C,G,T and *. 'Set Active Tags' brings up a menu which allows the user to select which tag types should be shown in the editor window. 'Set Output List' allows the user to create a list to which the names of readings can be added by clicking on them with a mouse button.

Traces for readings can be displayed by double clicking on their sequence or the consensus. The Contig Editor is identical to the Contig Joining Editor except for the Align button in the menu bar.

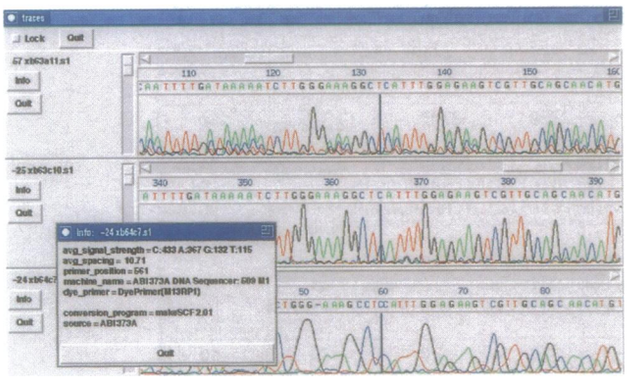

Figure 6. The Trace Display.

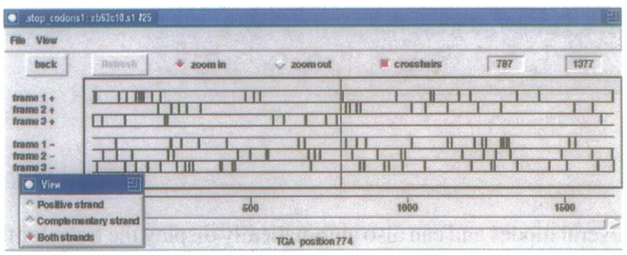

Figure 7. The Restriction Enzyme Map.

The Trace Display

When summoned from the editors (Fig. 6) each trace appears in its own scrollable window with its reading name and number in the top left corner. The vertical scale can be used to control the vertical magnification of the traces. The vertical black line in each display shows the position of the Contig Editor cursor. The individual traces can be scrolled separately, but if the 'Lock' button in the top left of the display is set, the the traces will all scroll in step with movement of the cursor in the Contig Editor. The Info buttons produce a window containing comments stored in the reading's SCF file.

The Stop Codon Map

The Stop Codon Map shown in Figure 7 is used to give an overview of the positions of all the stop codons on one or both strands of the consensus. If pairs of stop codons are clicked on in turn, their separation will be reported in the top right hand window. The 'Refresh' button is activated when a contig editor is operating on the same contig and, if the user clicks on it, the display will be automatically updated to reflect the current state of the consensus. That is, users can immediately see how changes made during editing affect the open reading frames in their sequence.

The Restriction Enzyme Map

This display is useful during sequence assembly projects when knowledge about restriction enzyme sites, either in the form of a map or simply as fragment sizes is available. The program produces displays like that shown in Figure 8. If the user clicks a mouse button on a pair of sites the program will display the separation of their cut sites in the top right hand box. The 'Edit'

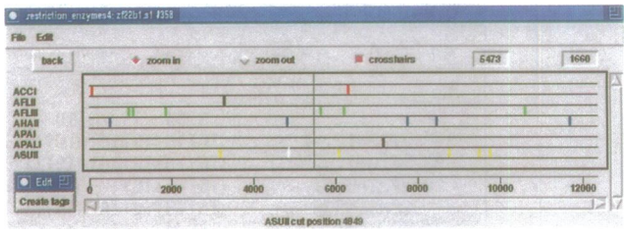

Figure 8. The Stop Codon Map.

menu contains a function to convert the cut sites for selected enzymes into tags. If this is done the contig will be labelled with landmarks that can help the user to orient the data and also to check the correctness of the assembly.

DISCUSSION

Minimising the cost of producing accurate consensus sequences is the overriding goal for sequence assembly software. Our previous paper in this journal (13) described how base calling accuracy estimates could be used to reduce editing time. The new program provides two further ways to reduce user time. First, the Experiment Suggestion functions automate what were previously interactive processes. Secondly, and most importantly, the new user interface, with its graphical displays, greatly simplifies many of the difficult operations that are necessarily still interactive. Another area that can be time-consuming is that of preparing the data for the assembly program, and our methods for automating this process are described elsewhere (14).

GAP includes a new Directed Assembly algorithm which has several possible applications. It can be used to assemble readings into known positions, possibly sequences produced as a result of the Experiment Suggestion functions or, more interestingly, it could be used to assemble all the data for projects using a directed sequencing strategy. At present GAP does not include a global assembly algorithm that can take advantage of the possibility that the majority of the data to cover a project are available prior to assembly. The Directed Assembly option is capable of inputting not only readings whose positions have been approximately mapped by an external program but, also, using Experiment File format, fully specified contigs consisting of already gapped readings. However we hope, either through the mechanism just outlined, or by an option callable from within GAP, to make a global assembly algorithm available in the near future.

Other areas of potential development include adding further analytical functions, such as gene detection methods, to assess the consensus during the project; the provision of new tools for the creation of sequence library ready files (the tags and Experiment Files are already a big step in this direction); and adding tools to improve the programs' utility for handling sequences obtained for diagnostic purposes. We believe that the structure we have now set in place put us in a good position for these further developments.

To find out how to obtain the programs contact Rodger Staden via electronic mail: rs@mrc-lmb.cam.ac.uk. Further informaiton can be obtained on the World Wide Web at http://www.mrc-lmb.cam.ac.uk/pubseq/.

ACKNOWLEDGEMENTS

We thank the many users of our package, particularly those at the Sanger Centre, for their comments, suggestions and bug reports. We are also grateful to Simon Dear for work done when a member of the group. We also thank Molly Craxton, Alan Bankier and Tony Crowther for critical reading of the manuscript. The work reported here was partly funded by an MRC Human Genome Mapping Program grant to R.S.

REFERENCES

1 Staden,R. (1982) *Nucleic Acids Res.*, **10**, 4731–4751.
2 Petola,H., Soderland,H. and Ukkonen,E. (1984) *Nucleic Acids Res.*, **12**, 307–321.
3 Dear,S. and Staden,R. (1991) *Nucleic Acids Res.*, **19**, 3907–3911.
4 Smith,S., Welch,W., Jakimcius,A., Dahlberg,T., Preston,E. and Van Dyke,D. (1993) *BioTechniques*, **14**, 1014–1018.
5 Huang,X. (1992) *Genomics*, **14**, 18–25.
6 Burks,C., Engle,M.L., Forrest,S., Parsons,R.J., Soderlund,C.A. and Stolorz,P.E. (1994) In J.C.Venter, M.D.Adams and C.Fields (eds), *Automated DNA Sequencing and Analysis*, Academic, London, pp. 249–259.
7 Lawrence,C.B., Honda,S., Parrot,N.W., Flood,T.C., Gu,L, Zhang,L., Mudita,J., Larson,S. and Myers,E.W. (1994) *Genomics*, **23**, 192–201.
8 Gleizes,A. and Henaut,A. (1994) *Comput. Applic. Biosci.*, **10**, 401–408.
9 Myers,E.W. and Miller,W. (1988) *Comput. Applic. Biosci.* **4**, 11–17.
10 Huang,X. (1994) *Comput. Applic. Biosci.*, **10**, 227–235.
11 Staden,R. (1984) *Nucleic Acids Res.*, **12**, 499–503.
12 Dear,S. and Staden,R. (1992) *DNA Sequence*, **3**, 107–110.
13 Bonfield,J.K. and Staden,R. (1995) *Nucleic Acids Res.*, **23**, 1406–1410.
14 Bonfield,J.K. and Staden, R. (1995) *DNA Sequence*, in press.
15 Pearson,W.R. (1994) In A.M. Griffin and H.G. Griffin, (eds), *Methods in Molecular Biology*, Humana Press., Totawa, NJ, **25**, 365–389.
16 Prober,J.M., Trainor,G.L., Dam,R.J., Hobbs,F.W., Robertson,C.W., Zugursky,R.J., Cocuzza,A.J., Jensen,M.A. and Baumeister,K. (1987) *Science*, **238**, 336–341.
17 Lee,L.G., Connell,C.R., Woo,S.L., Cheng,R.D., McArdle,B.F., Fuller,C.W., Halloran,N.D. and Wilson,R.K. (1992) *Nucleic Acids Res.*, **20**, 2471–2483.
18 Hillier,L. and Green,P. (1991) *PCR Methods Applic.*, **1**, 124–128.